6

NUMBER SYSTEMS

Positional Number System

Number system where the weight of a digit depends on its relative position with in the number, known is positional number system.

Table 6.1

Positional weighted Number systems	Base	Symbols (Digits)
Binary	2	0, 1
Ternary	3	0, 1, 2
Balanced Ternary	3	–1, 0, 1
Octal	8	0, 1, 2, 3, 4, 5, 6, 7
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Number systems are positional because the value of a symbol depends up on its position.

The number in the positional system can be represented as

$$\left(\underbrace{N_{n-1}N_{n-2}\ldots\ldots N_1N_0}_{\text{Integer part}} \overset{\text{Radix point}}{\cdot} \underbrace{N_{-1}N_{-2}\ldots\ldots N_{-m}}_{\text{Fractional part}}\right)_b$$

Here N_{n-1} is the Most Significant Digit (MSD) (extreme left digit)

N_{-m} is the Least Significant Digit (LSD) (extreme right digit)

b is the radix or base

n is the number of digit in the integerpart

m is the number of digit in the fractional part.

Problem

1. What is positional number system ? *(CU Nov. 18)*

Binary number system

Binary number system consists of only two symbols. i.e., 0 and 1. Its base is 2. The symbols used are called **bits**. Binary number system is a positional number system. The devices used in digital systems operate in ON or OFF states. i.e., the signals have two levels - zero or one. Zero is for LOW and one is for HIGH.

Weighting of Binary numbers

The right-most bit is LSB (Least Significant Bit) and left-most bit is MSB (Most significant Bit).

The wieight structure of a binary number is

$$2^{n-1}\ldots\ldots 2^3\,2^2\,2^1\,2^0 \cdot 2^{-1}\,2^{-2}\ldots\ldots 2^{-n}$$

Binary point

Why Binary system?

A switch can be either ON or OFF (1 or 0)

A transistor can be either cut-off or saturate state

A magnetic tape can be either magnetised or non magnetised

A signal can be either HIGH or LOW (1 or 0)

Note: 0 and 1 are called bits

1 Byte = 8 bits

1 K = 1024 $(= 2^{10})$

Binary to decimal conversion

Steps:

1. Write the binary number
2. Directly under the binary number write their respective weights working from right to left.
3. In case of fractions the weights of digits positions to the right of binary point are given by 2^{-1}, 2^{-2}, 2^{-3}, ... and so on.
4. Add the weights to obtain final result

Example : $(1\ 0\ 1\ 1\ 1)_2$ can be converted in to decimal from as

MSB (first bit), LSB (last bit)

$$\begin{array}{ccccccccc} 1 & & 0 & & 1 & & 1 & & 1 \\ \swarrow & & \swarrow & & \swarrow & & \swarrow & & \swarrow \\ 1\times 2^4 & + & 0\times 2^3 & + & 1\times 2^2 & + & 1\times 2^1 & + & 1\times 2^0 \\ 16 & + & 0 & + & 4 & + & 2 & + & 1 \end{array} = (23)_{10}$$

Example : 1 0 1 1 0 1 · 1 0 1

↑ Binary point

$$1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3}$$

$$= 32 + 0 + 8 + 4 + 0 + 1 + \frac{1}{2} + 0 + \frac{1}{8}$$

$$= (45.625)_{10}$$

Example : The binary number 1 0 1 0 1 · 0 1 1 can be written as

$$1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2} + 1\times 2^{-3}$$

$$= 16 + 0 + 4 + 0 + 1 + 0 + 0.25 + 0.125 = (21.375)_{10}$$

2. *Convert 111011.101_2 in to equivalent decimal number.*

(CU 2017 Nov.)

Decimal to Binary Conversion

Sum of weights method

Express the given number as a sum of powers of 2 and the units 1s and 0s in the appropriate digit positions. First find the maximum power of 2 and last minimum power of 2 (i.e., 2^0). In this way a list of seven binary weights would be 64, 32, 16, 8, 4, 2, 1 (i.e., $2^6, 2^5, 2^4, 2^3, 2^2, 2^1, 2^0$)

Example : Convert decimal number 29 in to binary

$$(29)_{10} = 2^4 + 2^3 + 2^2 + 2^0$$

$$= (16)_{10} + (8)_{10} + (4)_{10} + 0\times 2^1 + (1)_{10}$$

$$\therefore\ (29)_{10} = 1\ 1\ 1\ 0\ 1$$

$$(29)_{10} = (1\ 1\ 1\ 0\ 1)_2$$

Example : $(25.375)_{10} = 2^4 + 2^3 + 0\times 2^2 + 0\times 2^1 + 2^0 + 0\times 2^{-1} + 0.25 + 0.125$

There is a 0 in the 2^{-1} position

There is a 1 in the 2^{-2} position

There is a 1 in the 2^{-3} position

$$\therefore\ (25.375)_{10} = (1\ 1\ 0\ 0\ 1\cdot 0\ 1\ 1)_2$$

Repeated division by - 2 method

(Double Dabble Method)

A decimal number is repeatedly divided by 2 and the remainder is written on right side after each division. The remainders taken in reverse order form the binary number.

Example : Convert $(23)_{10}$ in to equivalent binary number.

2	23	1	(remainder)
2	11	1	(remainder)
2	5	1	(remainder)
2	2	0	,,
2	1	– 1	,,
	0		

$$\therefore\ (23)_{10} = (1\ 0\ 1\ 1\ 1)_2$$

Example : $(13)_{10}$

2	13	1 (LSB)
2	6	0
2	3	1
2	1	1 (MSB)
	0	

MSB (first bit), LSB (last bit)

$$(13)_{10} = (1\ 1\ 0\ 1)_2$$

Note: The first remainder is the LSB and the last remainder is the MSB.

Example : Convert $(186)_{10}$ in the equvalent binary number

		Remainder	
2	186	0	LSB ↑
2	93	1	
2	46	0	
2	23	1	
2	11	1	
2	5	1	
2	2	0	
2	1	1	MSB
	0		

$(186)_{10} = (1\,0\,1\,1\,1\,0\,1\,0)_2$

3. *The 8 bit binary equivalent of $(187)_{10}$ is ....* *(CU Nov. 20)*
   Ans : 10111011

Decimal fraction to binary

Repeatedly multiply by 2 and record any carriers in the integer position. Carries so obtained read downward form.

Example : $(0.6)_{10}$

$0.6 \times 2 = 1.2 = 0.2$ with a carry 1 ↓
$0.2 \times 2 = 0.4 = 0.4$ with a carry 0
$0.4 \times 2 = 0.8 = 0.8$ with a carry 0
$0.8 \times 2 = 1.6 = 0.6$ with a carry 1

$(0.6)_{10} = (.1001)_2$

Note: If more accuracy is needed, continue multiplying by 2 until you have as many digits as necessary for your application.

Example : $(0.32)_{10}$

$0.32 \times 2 = \mathbf{0}.64 = 0.64$ with a carry 0 ↓
$0.64 \times 2 = \mathbf{1}.28 = 0.28$ with a carry 1
$0.28 \times 2 = \mathbf{0}.56 = 0.56$ with a carry 0
$0.56 \times 2 = \mathbf{1}.12 = 0.12$ with a carry 1

$(0.32)_{10} = (.0101)_2$

Example : Convert $(25.6)_{10}$ ti its binary equivalent number

		Remainder
2	25	1 ↑
2	12	0
2	6	0
2	3	1
2	1	1
	0	

&

$0.6 \times 2 = 1.2$ 1 ↓
$0.2 \times 2 = 0.4$ 0
$0.4 \times 2 = 0.8$ 0
$0.8 \times 2 = 1.6$ 1

$(25.6)_{10} = (1\,1\,0\,0\,1 \cdot 1\,0\,0\,1)_2$

4. *The 8 bit binary equivalent of $(187)_{10}$ is ....* *(CU Nov. 17)*
   Ans : $(10\ 111011)_2$
5. *Convert $(48.25)_{10}$ to binary number* *(CU Nov. 17)*
6. *Convert 10 11.10 10 12 in to equivalent decimal number. Also convert 28.625_{10} in to binary number.* *(CU Nov. 20)*

Representation of integers

Positive and negative integers

Positive integers always have a sign bit (MSB) of 0.

Negative integers always have a sign bit (MSB) of 1.

7	6	5	4	3	2	1	0	← Bit position
	← Bit pattern representing the integer →							
MSB							LSB	

↑
Sign bit

0 – positive number

1 – Negative number.

The binary number which has 8 bits can be represented in memory as shown in fig. above.

Example : $(+23)_{10}$ is expressed in binary form as

$$\underbrace{0}_{\text{Sign bit}}\underbrace{0010111}_{\text{Magnitude bits}}$$

& $(-23)_{10}$ is expressed as

$$\underbrace{1}_{\text{Sign bit}}0010111$$

Positive numbers are stored in sign magnitude form whereas negative numbers are stored as 2's complements form.

For example : $(+3)_{10} \equiv (0011)_2$

For finding the binary value of $(-3)_{10}$ we have to find its 2's complement. How 2's complement is to be found?

$(+3)_{10} = (0011)_2$

1's complement = 1100

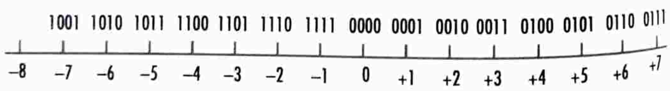
2's complement is obtained by adding 1 to 1's complement.

i.e., 2's complement = 1 1 0 0 +

$$\begin{array}{r} 1100 \\ +\ \ \ \ 1 \\ \hline 1101 \end{array}$$

$\therefore\ (-3)_{10} = (1\,1\,0\,1)_2$

Note: The following figure represents **decimal numbers** as **2's complements.**



–ve numbers in fig. are 2's complements of the +ve numbers.

i.e., Taking the 2's complement is equivalent to changing the sign of the number.

Sign magnitude form of positive numbers

Positive numbers are stored in sign magnitude form.

Example : Represent a decimal number $(29)_{10}$ in the memory of a 8-bit per word micro computer.

$(29)_{10} = (11101)_2$

The pattern for 29 has only five bits. In order to store this in 8 bits, expand this to 8 bits. For this add as many zeroes as are necessary towards left.

i.e., $(29)_{10} =$ 0 0 0 1 1 1 0 1

The left most bit (MSB) is 0.

So 29 is a positive number.

Similarly $(+23)_{10}$ is expressed as $\underbrace{0}_{\text{Sign bit}}$ 0 0 1 0 1 1 1

2's complement form of negative numbers

For representing high value negative numbers sign magnitude form can not be used. Here 2's complement form is used. Taking the 2's complement is equivalent to a sign change.

Example : Express the decimal number – 48 as an 8-bit number in the sign-magnitude, 1's complement and 2's complement forms.

Ans : 8 bit binary No. for + 48 is 0 0 1 1 0 0 0 0

In the sign-magnitude form, – 48 is produced by changing the sign bit to a 1 and leaving the magnitude bits as they are. The number is the magnitude bits as they are. The number is

1 0 1 1 0 0 0 0

In the 1's complement form – 48 is produced by taking the 1's complement of + 48 i.e., $(0\,0\,1\,1\,0\,0\,0\,0)_2$ as

1 1 0 0 1 1 1 1

In the 2's complement form, – 48 is produced by taking 2's complement of + 48 (0 0 1 1 0 0 0 0) as follows

00110000 ← represents + 48

11001111 ← 1^s complement

+1

11010000 ← 2^s complement.

Range of signed integer numbers

Example (i) Consider $(1)_{10}$ is a decimal number.

+ 1 can be written as 0 0 0 1 in 4 bit system.

– 1 can be written by taking 2's complement.

1's complement of 0 0 01 is 1 1 1 0

2's complement is 1 1 1 0 +

1

1 1 1 1

So $(-1)_{10}$ can be written as $(1\ 1\ 1\ 1)_2$

Similarly for $(2)_{10}$

+ 2 can be written as 0 0 1 0

– 2 can be written by taking 2's complement

1's complement is 1 1 0 1

2's complement 1 1 0 1 +

1

1 1 1 0

So $(-2)_{10}$ can be written as $(1\ 1\ 1\ 0)_2$

For $(7)_{10}$ binary equivalent is 0 1 1 1

$(-7)_{10}$ binary equivalent is 1 0 0 1

But for $(+8)_{10}$ → $(1000)_2$

$(-8)_{10}$ → $(1000)_2$

This is wrong.

So in the 4 bit representation + 7 is the largest +ve number and – 8 is the largest – ve number.

i.e., the largest –ve number has a magnitude that is one greater than the largest +ve number.

Similarly, using 8 bits the maximum +ve number is $(+127)_{10}$ = $(0111\ 1111)_2$ in 2's complement form and $(-128)_{10} = (1000\ 0000)_2$ is the largest negative number.

Decimal	Binary (8 bit numbers)
– 1	1000 0001
– 127	1111 1111
+ 1	0000 0001
+ 127	0111 1111

Over flow

We know, when adding or subtracting a number representing 2's complement, the resultant number has to lie within the range – 128 to + 127. If any number lies beyond this range there will be over flow in to the sign bit, causing a sign change.

For example : Add 110 and 60

Decimal number

1 1 0 + → 01111110 +

5 0 00110010

1 6 0 10100000

Here the sign bit is 1. So we are getting the sign bit for a –ve number. But we know that 160 is +ve. So the binary result also must be +ve. But we find a –ve number in binary. Here an overflow has produced. So the answer is incorrect.

Similarly if we add – 90 and – 87

Decimal number		2's complement
– 9 0 +	→	1 0 1 0 0 1 1 0 +
– 8 7	→	1 0 1 0 1 0 0 1
– 1 7 7		$\boxed{1}$ 0 1 0 0 1 1 1 1

Carry

Here the 8 - bit answer is 0 1 0 0 1 1 1 1. The sign bit is 0. So this represents a +ve number. But we are expecting a –ve number. So overflow has produced. So the answers not correct.

In briefly in using 8 bits we are able to handle only numbers in between + 127 to – 128. By handling numbers above and below of + 127 & – 128 overflow takes place.

Overflows are a software problem.

For 2^s complement signed numbers, the range of value for n-bit numbers is from $-(2^{n-1})$ to $+(2^{n-1}-1)$

Example : For 8 bit numbers.

Range is from $-(2^7) = -128$ to $+(2^7-1) = 127$

By representing the decimal numbers in binary form using eight bits the maximum +ve number is $(127)_{10} = (0\,1\,1\,1\,1\,1\,1\,1)_2$ and in 2's complement form $(-128)_{10} = (1\,0\,0\,0\,0\,0\,0\,0)_2$ is the largest negative number.

For 2^s complement signed numbers, the range of value for n-bit numbers is from $-(2^{n-1})$ to $+(2^{n-1}-1)$

Similarly in the 4 bit representation +7 is the largest positive number and – 8 is the largest negative number.

By writting positive and negative binary equivalents for decimal numbers 1 to 8 you will realise this. For 8, the binary equivalent is 1000. Sign bit is 1. It is a negative number. So it is not acceptable because 8 is a +ve number.

Decimal value of signed numbers

(i) Sign-magnitude

Example : Determine the decimal value of the signed binary number expressed in sign magnitude: 1 0 0 1 0 1 0 0

$$\underbrace{1}_{\text{Sign bit}}\ \underbrace{0010100}_{\text{7 magnitude bits}}$$

Write the seven magnitude bits and their power of two weights

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	0	0

Summing the weights where there are 1^s

$1 \times 2^4 + 1 \times 2^2 = 16 + 4 = 20$

The sign bit is 1

∴ The decimal number is $(-20)_{10}$

Example : Determine the decimal value of the sign magnitude number 0 1 1 1 0 1 1 1

$$\underbrace{0}_{\text{Sign bits}}\ \underbrace{1110111}_{\text{Magnitude bits}}$$

$$1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$
$$= 64 + 32 + 16 + 4 + 2$$
$$= 119$$

The Sign bit is 0

∴ The decimal number is $(+119)_{10}$

(ii) 1^s complement

Decimal values of +ve number in the 1^s complement form are found by summing the weights of all bits positions.

Decimal values of –ve numbers are found by assigning a –ve value to the weight of the sign bit (8^{th} bit) summing all the weights and adding 1 to the result.

Example : Determine the decimal value of the 1^s complement number 1 1 1 0 1 0 1 1

1 1 1 0 1 0 1 1 is the binary number

This is a –ve number since sign bit is 1

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	1	0	1	0	1	1

$- 128 + 64 + 32 + 8 + 2 + 1 = - 21$

add 1 to the result

$-21 + 1 = \underline{\underline{-20}}$

(iii) 2^s complement

Here decimal values of +ve and –ve numbers are found by summing the weights in all positions. The weight of the sign bit in a –ve is given a –ve value.

Example : Determine the decimal value of the 2^s complement number 1 1 0 1 0 1 1 1

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	1	0	1	1	1

$- 128 + 64 + 0 + 16 + 0 + 4 + 2 + 1$

$- 128 + 64 + 16 + 4 + 2 + 1 = \underline{\underline{-41}}$

2^s complement method is simple for presenting a signed integer number.

BCD representation

BCD stands for Binary Coded Decimal

Here each decimal number (0 through 9) are converted in to binary codes by replacing each decimal digit with a string of 4 binary digits called nibbles.

8421 BCD code

The binary weights of the dour bits ($2^3, 2^2, 2^1, 2^0$) are indicated by 8, 4, 2, 1.

Table 6.2

BCD	Decimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

***Note*:** only the above codes for up to 9 are used. The other six codes (1010, 1011, 1100, 1101, 1110 & 1111) are not used since it is 8421 code. (8 is maximum).

Example : Convert the decimal number 9673 to BCD.

$$\begin{array}{ccccl} 9 & 6 & 7 & 3 & \longrightarrow \text{Decimal} \\ \downarrow & \downarrow & \downarrow & \downarrow & \\ (1001 & 0110 & 0111 & 0011) & \longrightarrow \text{BCD} \end{array}$$

BCD Addition

Step 1: Add two BCD number using ordinary binary addition

Step 2: If four -bit sum is equal to 0 or less than 9, no correction is needed. it is a valid BCD number

Step 3: If four bit sum is greater than 9 or if a carry is generated from the four-bit sum, the sum is invalid.

Step 4: To correct the invalid sum, add $(6)_{10}$ i.e., $(0110)_2$ to the four bit sum. If a carry results from this addition, add it to the next higher order BCD digit.

Problem: (1) Add $(35)_{10}$ and $(13)_{10}$ in the 8421 code

35	→	0 0 1 1 0 1 0 1	(35 in BCD)
13	→	0 0 0 1 0 0 1 1	(13 in BCD)
48		0 1 0 0 1 0 0 0	

Here there is no carry, no illegal code. So the answer is correct.

(2) Add $(479.6)_{10}$ and $(536.8)_{10}$ in 8421 code

679.6	→	0110	0111	1001	.	0110
+ 536.8	→	+ 0101	0011	0110	.	1000
1216.4		1011	1010	1111	.	1110

All are illegal codes or invalid. So we have to add $(6)_{10}$ or $(0110)_2$ to each.

∴					
		1011	1010	1111	· 1110
		+ 0110	0110	0110	· 0110
Propagate carry		$\boxed{1}$ 0001	$\boxed{1}$ 0000	$\boxed{1}$ 0101	$\boxed{1}$ · 0100
	+1 ←	+1 ←	+1 ←	+1 ←	
	0001	0010	0001	0110	· 0100
	↓	↓	↓	↓	↓
	1	2	1	6	.4

Now we arrive at the corrected sum $(1216.4)_{10}$

Example : $(435)_{10}$ can be written in BCD as 0 1 0 0 0 0 1 1 0 1 0 1

disadvantage : It requires too much space.

Representation of real numbers

Real number consists of an integer part and a fraction part together with a positive or negative sign.

Example ; 15.3, 126.78, – 0.734 etc. are the real numbers in decimal system.

1 0 1 1 . 0 1, 1 1 1 0 1. 0 1 1 etc. are the real numbers in binary system.

Example : 1 0 1 1 0 1 · 1 0 1

↑ Binary point

$$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

$$= 32 + 0 + 8 + 4 + 0 + 1 + \frac{1}{2} + 0 + \frac{1}{8}$$

$$= (45.625)_{10}$$

Example : The binary number 1 0 1 0 1 · 0 1 1 can be written as

$$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$$

$$= 16 + 0 + 4 + 0 + 1 + 0 + 0.25 + 0.125 = (21.375)$$

7. *Convert $(111011.101)_2$ in to equivalent decimal number* *(CU Nov. 17)*

Example : Find the binary equivalent of the decimal fraction 0.625.

Ans:

Table 6.3

Fraction	Fraction × 2	Integer part	Remainder
0.625	1.250	1	0.25
0.25	0.50	0	0.50
0.50	1.00	1	.00
			Answer $(101)_2$

Example : Find the binary equivalent of $(13.625)_{10}$

Ans: $(1101.101)_2$

Floating point numbers

Floating point number consist of -

(i) Mantissa (ii) exponent (iii) a sign.

– Mantissa is the magnitude of the number

(It is between 0 and 1)

– Exponent is the number of places that the decimal or binary point is to be moved.

Example : 242 506 800

Here mantissa is 0.24 25068 Exponent is 9

The decimal point is moved to left of all digits so that mantissa becomes in between 0 and 1.

∴ The floating point number is

$$0.242\ 506\ 800 \times 10^9$$

Example : Convert the following in the normalized floating point notation.

(i) $13.2 \times 10^8 \quad = 0.132 \times 10^{10}$

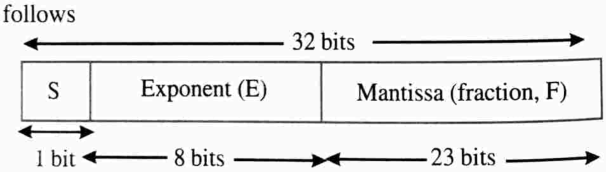
(ii) $0.0152 \times 10^{19} = 0.152 \times 10^{18}$

(iii) $1.91 \times 10^{-29} \quad = 0.191 \times 10^{-28}$

Note: In general, normalized floating point numbers are of the form $m \times b^n$ where m is the mantissa which satisfies the relation $\frac{1}{b} < m < 1$ and b is the base of the number system and n the exponent.

Representation of floating point binary numbers

We consider a device that accepts 32-bit floating point numbers as follows



Binary Arithmetic

Binary addition

Binary addition is performed in the same manner as decimal addition.

Table 6.4

Augend	+	Addend	Carry (C)	Sum (S)	Result
0	+	0	0	0	0
0	+	1	0	1	1
1	+	0	0	1	1
1	+	1	1	0	10

Example : Find 1 + 1 + 1

1 +
1
10 +
1
11 Sum 1, Carry 1

Example : Find 1 + 1 + 1 + 1

1 +
1
10 +
1
11 +
1
100

Example : Add binary numbers 1 0 0 1 and 1 1 1 0

Binary		Decimal
1 0 0 1 +	≡	9 +
1 1 1 0		14
1 0 1 1 1		23

Example : Add 1 0 . 0 0 1 and 1 1 . 1 1 0

$$\begin{array}{r} 10.001\ + \\ 11.110 \\ \hline (101.111)_2 \\ \hline\hline \end{array} \qquad \begin{array}{r} 2.125\ + \\ 3.750 \\ \hline (5.875)_{10} \\ \hline\hline \end{array}$$

Example : Add 1 1 1 1 0 0 1 and 1 1 0 0 1 0 1

$$\begin{array}{r} 1\quad\ 1\quad\ 1\ \leftarrow \text{Carry} \\ 1111001\ + \\ 1100101 \\ \hline 11011110 \\ \hline\hline \end{array}$$

Binary Subtraction

Subtraction is the inverse operation of addition

Table 6.5

Minuend	–	Subtrahend	Result
0	–	0	0
0	–	1	0 with a borrow of 1
1	–	0	1
1	–	1	0
10	–	1	1

Note: To subtract, it is necessary to maintain procedure for subtracting a large digit from a small. The only case in which this occurs with binary numbers is when 1 is subtracted from 0. The remainder is 1, but it is necessary to borrow 1 from the next column to the left.

Example :

Binary	Decimal
1 0 1	5
– 0 1 0	– 2
0 1 1	3

Laws
0 – 0 = 0
1 – 0 = 1
1 – 1 = 0
0 – 1 = 0 with a borrow of 1 10 – 1 = 01

Here

Step 1: In first column, 1 – 0 = 1

Step 2: In second column, 0 – 1

(here taken borrow 1 from next left column)

∴ 10 – 1 = 1

Step 3: In third column

1 is already borrowed

a 0 is left

0 – 0 = 0

Example : (i)

$$\begin{array}{r} 1001 \\ -100 \\ \hline 101 \\ \hline\hline \end{array}$$

(ii)

$$\begin{array}{r} 110.01 \\ -100.1 \\ \hline 1.11 \\ \hline\hline \end{array}$$

(iii)

$$\begin{array}{r} 101 \\ -011 \\ \hline 010 \\ \hline\hline \end{array}$$

In example : (iii)

Step 1: 1 – 1 = 0 (Right column)

Step 2: 0 – 1 Borrow 1 from next column to the left

∴ 10 – 1 = 1

Step 3: (Left most column)

1 is borrowed to right ∴ 0 is there

0 – 0 = 0

Example :

$$\begin{array}{r} 10000 \\ -011 \\ \hline (1101)_2 \end{array} \equiv \begin{array}{r} (16)_{10} \\ -(3)_{10} \\ \hline (13)_{10} \end{array}$$

Binary Multiplication

Rules:

$$\boxed{\begin{array}{l} 0 \times 0 = 0 \\ 0 \times 1 = 0 \\ 1 \times 0 = 0 \\ 1 \times 1 = 1 \end{array}}$$

The binary multiplication involves forming the partial products, shifting each successive partial product left to one place, and then adding all the partial products.

Example : (i) 11 × 11

$$\begin{array}{r} 11 \\ \times\ 11 \\ \hline 11 \\ +\ 11 \\ \hline 1001 \end{array} \qquad \begin{array}{r} (3)_{10} \\ \times\ (3)_{10} \\ \hline (9)_{10} \end{array}$$

(ii) 100 × 10

$$\begin{array}{r} 100 \\ \times\ 10 \\ \hline 000 \\ +100 \\ \hline 1000 \end{array}$$

(iii) 1110 × 1101

$$\begin{array}{r} 1110 \\ \times 1101 \\ \hline 1110 \\ +\ 0000 \\ +1110 \\ 1110 \\ \hline 10110110 \end{array}$$

(iv) 1.01 × 10.1

$$\begin{array}{r} 1.01 \\ \times\ 10.1 \\ \hline 101 \\ 000 \\ 101 \\ \hline 11.001 \end{array} \equiv \begin{array}{r} (1.25)_{10} \\ \times (2.5)_{10} \\ \hline (3.125)_{10} \end{array}$$

Binary division

Rules:

$$\boxed{\begin{array}{l} 0 \div 1 = 0 \\ 1 \div 1 = 1 \\ 0 \div 0 = \text{no meaning} \\ 1 \div 0 = \text{no meaning} \end{array}}$$

Assume the dividend is no larger than divisor.

Steps: (i) Start from the left on the dividend

(ii) A series of subtractions are performed in which the divisor is subtracted from the dividend.

(iii) If subtraction is possible put a 1 in the quotient and subtract the divisor for the corresponding digits of dividend.

(iv) If subtraction is not possible (i.e., divisor greater than remainder), put a 0 in the quotient.

(v) Bring down the next digit to add to remainder digits.

Example : (i) $(1000)_2 \div (10)_2 \equiv (8) \div (2)_{10} = (4)_{10}$

$$\begin{array}{r} 10)\,1000\,(100 \\ 10 \\ \hline 000 \end{array}$$

(ii) $(11001)_2 \div (101)_2 \equiv (25)_{10} \div (5)_{10}$

$$\begin{array}{r} 101)\ 11001\ (101 \\ -101 \\ \hline 0101 \\ -101 \\ \hline 000 \\ \hline \end{array} \equiv \frac{25}{5} = (5)_{10}$$

Complements and its Algebra

1^s & 2^s complements of Binary Numbers

1^s complement

1^s complement of any binary number is obtained by changing each 1 in the number to a 0 and each 0 in the number to a 1.

Example : Find 1^s complement of the following number.

Binary number	1^s complement
11 10	0001
101 101	010010
1100 1110 1011 1001	0011 0001 0100 0110

1^s complement Arithmetic

(a) Subtrahend is smaller than the minuend

Steps: (i) Complement the subtrahend (by converting 1^s into 0^s & Viceversa)

(ii) Proceed as in addition

(iii) Disregard the carry and add 1 to the total (emd-around-carry)

Example :

$$\begin{array}{r} (111001010)_2 \\ -(110110101)_2 \\ \hline \end{array} \rightarrow \begin{array}{rl} 111001010 & \\ +001001010 & (1\text{'s complement}) \\ \hline 1000010100 & \\ 1 & (\text{end around carry}) \\ \hline 000010101 & \\ \hline \end{array}$$

(b) If subtrahend is larger than the minuend

Steps: (i) Complement the subtrahend

(ii) Proceed as in addition

(iii) Complement the result

(iv) Place a –ve sign infront of the result

Example :

$$\begin{array}{r} 1101011 \\ -1110101 \\ \hline \end{array} \rightarrow \begin{array}{rl} 1101011 & \\ +0001010 & (1^s \text{ complement}) \\ \hline 1110101 & \end{array}$$

Complement the result → 0001010

Place –ve sign → $\underline{\underline{-0001010}}$

2's Complement

2^s complement of a binary number is found by adding 1 to the LSB of the 1^s complement. 2^s complement = (1^s complement) + 1.

Example : Write 2^s complement of the following binary numbers

(i) 1011

Ans:

$$\begin{array}{rl} 0100 & \leftarrow 1^s \text{ complement of } 1\,0\,1\,1 \\ +1 & \leftarrow \text{Add } 1 \\ \hline 0101 & \leftarrow 2^s \text{ complement} \\ \hline \end{array}$$

(ii) 1 0 1 1 0 1 1 0

Ans: 0 1 0 0 1 0 0 1 ← 1's complement
+ 1 ← Add 1
0 1 0 0 1 0 1 0 ← 2's complement

Example : Find 2's complement of the decimal number $(5)_{10}$

0 1 0 1 ← Binary number form

1 0 1 0 ← 1's complement
+ 1 ← Add 1
1 0 1 1 2's complement.

Alternate method for finding 2's complement

Step 1: Start at the right with the LSB and write the bits as they are up to and including the first 1.

Step 2: Take the 1's complement of the remaining bits.

Example : 1 1 0 0 1 0 0 0 ← Binary number

0 0 1 1 1 0 0 0

1's complement of original bits; These bits stay the same

∴ Ans: 0 0 1 1 1 0 0 0 ← (2's complement)

Arithmetic Operations with Signed Numbers

Addition

Case (i) When both numbers are positive

Example :

0 0 0 0 1 0 1 1	1 1
+ 0 0 0 0 0 1 0 1	+ 5
0 0 0 1 0 0 0 0	$(16)_{10}$

Case (ii) Addend is +ve and augend is –ve (+ve number with magnitude larger than –ve number)

Example :

2's Complement of – 4

0 0 0 0 1 1 0 1	1 3
+ 1 1 1 1 1 1 0 0	+ – 4
$\boxed{1}$ 0 0 0 0 1 0 0 1	9

Discard carry

The final carry bit is discarded.

Case (iii) A negative number with magnitude larger than that of +ve number.

2's Complement of – 13

0 0 0 0 0 1 0 0	+ 4
1 1 1 1 0 0 1 1	+ – 13
1 1 1 1 0 1 1 1	– 9

No carry

Case (iv) both number –ve

$(4)_{10} = (0\,1\,0\,0)_2$

2's Complement of – 4

1 1 1 1 1 0 1 1	– 4
1	+ – 8
1 1 1 1 1 1 0 0	– 12

Similarly,

$(8)_{10} = (1\,0\,0\,0)_2$ = $(0\,0\,0\,0\,1\,0\,0\,0)_2$

2's Complement of – 8 1 1 1 1 0 1 1 1
+ 1
1 1 1 1 1 0 0 0

Adding

1 1 1 1 1 1 0 0
1 1 1 1 1 0 0 0
$\boxed{1}$ 1 1 1 1 0 1 0 0

Discard carry

Note: add the two number and discard any final carry bit.

Overflow

While adding or subtracting a number representing 2^s complement, the resultant number should lie with in the range -128 to $+127$. Whenever, it lies beyond this range there is overflow into the sign bit, causing a sign change.

When two +ve or –ve numbers are added then their sum may be outside the range –128 to + 127.

Example : Add 100 and 50

100	0 1 1 0 0 1 0 0
50	0 0 1 1 0 0 1 0
$(150)_{10}$	$\boxed{1}$ 0 0 1 0 1 1 0

Sign bit

We have added two + ve numbers. We expect a + ve result. But due to MSB = 1, the number becomes –ve. Here an overflow has provided resulting an incorrect answer.

Example : Add decimal number – 97 and – 85

	2^s complement
– 97	1 0 0 1 1 1 1 1
– 85	1 0 1 0 1 0 1 1
– 182	$\boxed{1}$ 0 1 0 0 1 0 1 0

Carry

Here there is an overing flow. Eight bit answer is 0 1 0 0 1 0 1 0.

The sign bit of this is 0. It represents a +ve number. But number is –ve.

Note: Overflow is a software problem. Programme must test for an overflow after each addition or subtraction.

Numbers added two at a time

Example : Add the signed number:

0 1 0 0 0 1 0 1, 0 0 0 1 1 0 1 0, 0 0 0 0 1 1 1 0 and 0 0 0 1 0 0 1 1

69	0 1 0 0 0 1 0 1
+ 26	+ 0 0 0 1 1 0 1 0
95	0 1 0 1 1 1 1 1
+ 14	+ 0 0 0 0 1 1 1 0
109	0 1 1 0 1 1 0 1
19	+ 0 0 0 1 0 0 1 1
128	1 0 0 0 0 0 0 0

Note: First two numbers are added. To the sum of these the third number is added. To the result fourth number added and so on.

Subtraction

Subtraction is another form of addition. Consider we have to subtract + 5 (Subtrahend) from + 8 (the minuend). This is equivalent to adding – 5 to + 8.

So in subtraction process, the sign of the subtrahend is changed. Then it is added to minuend (For changing the sign of a binary number 2's complement is taken).

Example : $(+5)_{10}$ = $(0\,0\,0\,0\,0\,1\,0\,1)_2$

1^s complement		1 1 1 1 1 0 1 0	
		+ 1	
2's complement		$(1\,1\,1\,1\,1\,0\,1\,1)_2$	$= (-5)_{10}$
$(+8)_{10}$	=	$(0\,0\,0\,0\,1\,0\,0\,0)_2$	
∴ $-5+8$	=	1 1 1 1 1 0 1 1	
	+	0 0 0 0 1 0 0 0	
		$\boxed{1}$ 0 0 0 0 0 0 1 1	$= (+3)_{10}$

Discard carry

Example : Subtract 0 1 0 0 0 1 1 1 from 0 1 0 1 1 0 0 0

i.e., $(71)_{10}$ from $(88)_{10}$

i.e., $(88)_{10} - (71)_{10}$, i.e., $(88)_{10} + (-71)_{10}$

```
           0 1 0 1 1 0 0 0  (Minuend + 88)
         + 1 0 1 1 1 0 0 1  [2's complement of subtrahend (– 71)10]
Discard [1] 0 0 0 1 0 0 0 1  = (+ 17)10
carry
```

Multiplication

Multiplication is possible by addition. In direct addition method - add the multiplicand a number of times equal to the multiplier.

For Ex: 5×3

5 is the multiplicand

3 is the mutliplier

$\therefore\ 5 + 5 + 5 = 15$

Example: Multiply 0 1 1 0 0 0 0 1 by 0 0 0 0 0 1 1 0 using the direct addition method.

Ans: Here $(0\,0\,0\,0\,0\,1\,1\,0)_2 = (+6)_{10}$ is the mutliplier. 0 1 1 0 0 0 0 1 is the multiplicand. So multiplicand is added six times.

0 1 1 0 0 0 0 1	1st time
+ 0 1 1 0 0 0 0 1	2nd time
1 1 0 0 0 0 1 0	Partial sum
+ 0 1 1 0 0 0 0 1	3rd time
1 0 0 1 0 0 0 1 1	Partial sum
+ 0 1 1 0 0 0 0 1	4th time
1 1 0 0 0 0 1 0 0	Partial sum
+ 0 1 1 0 0 0 0 1	5th time
1 1 1 1 0 0 1 0 1	Partial sum
+ 0 1 1 0 0 0 0 1	6th time
1 0 0 1 0 0 0 1 1 0	Product

Verification: $(97)_{10} \times (6)_{10} = (582)_{10}$

Note: Here the sign bit of the multiplicand is 0. So it has no effect on result. All of he bits in the product are magnitude bits.

Other number systems

Hexadecimal Numbers

It has a base of 16. It requires 16 symbols. These are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. it is also called **alphanumeric** number system.

Table 6.6

Decimal (Radix 10)	Binary (Radix 2)	Hexadecimal (Radix 16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	0001 0000	10
17	0001 0001	11
18	0001 0010	12
19	0001 0011	13
20	0001 0100	14

Hexadecimal number represents group of four binary digits. These are used in computer and microprocessor applications.

Binary to Hexadecimal conversion

Break the binary number in to 4-bit groups starting from right most bit. Replace each 4-bit group with the equivalent hexadecimal number.

Example : Convert $[1\,0\,0\ \ 1\,1\,1\ \ 0\,0\ \ 1\,0\,1]_2$ to hexadecimal number.

Ans : Grouping the number in to 4-bits

$$\underbrace{0\,1\,0\,0}_{4}\qquad \underbrace{1\,1\,1\,0}_{E}\qquad \underbrace{0\,1\,0\,1}_{5}$$

$\therefore\quad [1\,0\,0\,1\,1\,1\,0\,0\,1\,0\,1]_2 = [4\,E\,5]_{16} = [4\,E\,5]_H$

Problem: Convert the binary number

1 0 0 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0 0 to hexadecimal

$$\text{Ans:}\quad \underbrace{0\,1\,0\,0}_{4}\qquad \underbrace{1\,1\,1\,1}_{F}\qquad \underbrace{0\,1\,1\,1}_{7}\qquad \underbrace{1\,0\,0\,1}_{9}\qquad \underbrace{1\,1\,0\,0}_{C}$$

$$= (4\,F\,7\,9\,C)_{16}$$

Problem: Convert $[1\,1\,1\,1\,0\ \ 1\,1\,1\,0\,1\,1\,1\,.\,1\,1\,1\,0\,1\,0]_2$ to hexadecimal number.

$$\underbrace{1\,1\,1\,1}_{F}\qquad \underbrace{0\,1\,1\,1}_{7}\qquad \underbrace{0\,1\,1\,1}_{7}\ \ .\ \ \underbrace{1\,1\,1\,0}_{E}\qquad \underbrace{1\,0\,0\,0}_{8}$$

$$= (F\,77\cdot E8)_{16}$$

Hexadecimal-to-Binary Conversion

Each digit is converted in to a group of 4 binary bits. (reverse process of earlier)

Problem: Convert 6 B D 3 to binary

$$\overbrace{(0\,1\,1\,0}^{6}\qquad \overbrace{1\,0\,1\,1}^{B}\qquad \overbrace{1\,1\,0\,1}^{D}\qquad \overbrace{0\,0\,1\,1)_2}^{3}$$

Problem: Convert $[2\,A\,B\,.\,82]_{16}$ to binary number

$$\begin{array}{lllllll} & [2 & A & B & . & 8 & 2]_4 \\ = & [0\,0\,1\,0 & 1\,0\,1\,0 & 1\,0\,1\,1 & . & 1\,0\,0\,0 & 0\,0\,1\,0]_{16} \end{array}$$

8. *The binary equivalent of the hexadecimal number EF is ....*

Ans : 1110 1111 *(CU Nov. 20, CU Nov. 18)*

(Verification : $\text{Ex }16' + F \times 16^{o} = 14 \times 16 + 15 \times 1 = 224 + 15 = 239$)

Hexadecimal to Decimal Conversion

Method (i) Convert the Hex. number to binary. Then convert to decimal.

Example : Convert $(6\,B\,D)_H$ to binary

$$\begin{array}{llll} \textit{Ans}: & 6 & B & D \\ & (0\,1\,1\,0 & 1\,0\,1\,1 & 1\,1\,0\,1)_2 \end{array}$$

Method (ii) Convert $(B\,6\,C)_H$ in to decimal

Ans: $(B\,6\,C)_H = B \times 16^2 + 6 \times 16^1 + C \times 16^0$

$= 11 \times 256 + 6 \times 16 + 12 \times 1$

$= 2816 + 96 + 12 = (2924)_{10}$

Problem: Convert $[4\,A\,B \cdot 6]_H$ to decimal

Ans: $[4AB.6]_H = 4 \times 16^2 + A \times 16^1 + B \times 16^0 + 6 \times 16^{-1}$

$= 4 \times 256 + 10 \times 16 + 11 \times 1 + 6 \times 0.0625$

$= 1024 + 160 + 11 + 0.3750$

$= [1195.4]_{10}$

Problem: Convert $[A\,F \cdot 3\,C]_{16}$ to decimal

$[A\,F.3\,C]_H = A \times 16^1 + F \times 16^0 + 3 \times 16^{-1} + C \times 16^{-2}$

$= 10 \times 16 + 15 \times 1 + 3 \times 16^{-1} + 12 \times 16^{-2}$

$= 160 + 15 + 0.1875 + 0.046875$

$= (175.23)_{10}$

Decimal to Hexadecimal Conversion

Method (i): divide by 16 until zero is obtained in the quotient. The remainders can then be written from bottom to top to obtain the hexadecimal digits.

Example : Convert $(866)_{10}$ to hexadecimal

$866 \div 16 = 54$ + with a remainder of 2

$54 \div 16 = 3$ + with a remaider of 6

$3 \div 16 = 0$ + with a remainder of 3

$\therefore \quad (866)_{10} = (362)_{16}$

Method (ii) : when a quotient has a fractional part, the fractional part is multiplied by the divisor to get the remainder.

Example : Convert 2591 to hexadecimal

$\frac{2591}{16} = 161.9375 \rightarrow 0.9375 \times 16 = 15 = F$

$\frac{161}{16} = 10.0625 \rightarrow 0.0625 \times 16 = 1 = 1$

$\frac{10}{16} = 0.625 \rightarrow 0.625 \times 16 = 10 = A$

stop when whole number quotient is zero

$\therefore \quad (2591)_{10} = [A\ 1\ F]_H$

Problem: Convert $[106.0664]_{10}$ to hexadecimal

Integer part

		Hex.
$\frac{106}{16}$	= 6 + remainder of 10	A
$\frac{6}{16}$	= 0 + remainder of 6	6

Fractional part

			Hex.
$0.0664 \times 16 = 1.0624$	$= 0.0624$ +	with a carry of	1 (1)
$0.0624 \times 16 = 0.9984$	$= 0.9984$ +	with carry of	0 (0)
$0.9984 \times 16 = 15.9744$	$= 0.9744$ +	with carry of	15 (F)
$0.9744 \times 16 = 15.5904$	$= 0.5904$ +	with carry of	15 (F)

$\therefore [106.0664]_{10} = [6A.10FF]_{16}$

Note: For integer part remainders can be written from bottom to top and for fractional part remainders can be written from top to bottom.

9. *Convert decimal number 378 to a 16 bit number by first converting to hexadecimal* *(CU Nov. 19)*

Hexadecimal addition

Points: (i) think hexadecimal digits in terms of decimal values, if possible.

Example : $(0 \text{ to } 9)_{16}$ directly as $(0 \text{ to } 9)_{10}$

A, B, C, D, E, F, as $(10, 11, 12, 13, 14, 15)_{10}$

(ii) If sum is less than $(15)_{10}$ bring corresponding hexadecimal

(iii) If sum is greater than $(15)_{10}$ bring down the amount of the sum that exceeds 16_{10} and carry a 1 to next column.

Example : (a) $(42)_{16} + (36)_{16}$

Left	Right
4	2
+ 3	6
$(7\ 8)_{16}$	

Right column $2_{16} + 6_{16} = 2_{10} + 6_{10} = 8_{10} = 8_{16}$

Left column $4_{16} + 3_{16} = 4_{10} + 3_{10} = 7_{10} = 7_{16}$

(b) $(68)_{16} + (32)_{16}$

Left	Right
6	8
3	6
$(9\,E)_{16}$	

Right column $8_{16} + 6_{16} = 8_{10} + 6_{10} = 14_{10} = E_{16}$

Left column $6_{16} + 3_{16} = 6_{10} + 3_{10} = 9_{10} = 9_{16}$

(c) $EF_{16} + BC_{16}$

Left	Right
E	F
B	C

Right column

$F_{16} + C_{16} = 15_{10} + 12_{10} = 27_{10}$

$27_{10} - 16_{10} = 11_{10} = B_{16}$ with carry 1

Left column

$E_{16} + B_{16} + 1_{16} = 14_{10} + 11_{10} + 1_{10} = (25)_{10}$

$25_{10} - 16_{10} = 9_{10} = 9_{16}$ with a 1 carry.

To get the 2's complement of a hexadecimal number

Hints: – Convert Hexadecimal to Binary

– Take 2^s complement

– Convert the result to hexadecimal.

Example : $(3A)_{16} \rightarrow (0011\ 1010)_2 \rightarrow 11000110 \rightarrow (C6)_{16}$

Hexadecimal subtraction

– Subtract the Hexadecimal number from the maximum hexadecimal number

– add 1

Hexadecimal → Subtract from Maximum FF → To get 1s complement in hexadecimal, add + 1 → 2's complement in hexadecimal

Example :

2A → 01 2 3 456789 A BCD EF / FE D C BA9876 5 43210 → D 5 + 1 → D 6

Octal numbers

A number system that uses eight digits 0, 1, 2, 3, 4, 5, 6, 7. It has a base of eight.

Octal to decimal conversion

Example : $[6427.35]_8$

$= 6 \times 8^3 + 4 \times 8^2 + 2 \times 8^1 + 7 \times 8^0 + 3 \times 8^{-1} + 5 \times 8^{-2}$

$= 3072 + 256 + 16 + 7 + 0.375 + 0.0781$

$= [3351.4531]_{10}$

Decimal to Octal conversion

(i) Repreated division by 8 method.

Example : Convert $(429)_{10}$ to octal number

Remander

$\frac{429}{8} = 53.625 \rightarrow 0.625 \times 8 = 5$

$\frac{53}{8} = 6.625 \rightarrow 0.625 \times 8 = 5$

$\frac{6}{8} = 0.75 \rightarrow 0.75 \times 8 = 6$

Stop when whole number quotient is zero

$= (6\ 5\ 5)_8$

MSD LSD

Binary to Octal Conversion

(i) Binary-triplet method

The binary digits are grouped into groups of three on each side of the binary point with zeros added on eitherside if needed. (The highest digit in the octal system is 7).

Example : Convert $[1\,0\,1\,1\,0\,.\,1\,0\,1\,0\,1\,0]_2$ to octal

Ans: Step (i): Divide in to groups of three bits

$$\begin{array}{cccc} 010 & 110 & 101 & 010 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 2 & 6 & 5 & 2 \end{array}$$

$$= (26.52)_8$$

Octal to binary conversion

Example : $(647)_8$ to binary

$$\begin{array}{lccc} \text{Ans: Octal digit} & 6 & 4 & 7 \\ & \downarrow & \downarrow & \downarrow \\ \text{Binary} & 110 & 100 & 111 \end{array}$$

$$(647)_8 = (110\ 100\ 111)_2$$

10. Convert $(B2F)_{16}$ to octal *(CU Nov. 19)*

Exercises

1. Explain 1's complement method of binary subtraction with example.
2. Represent the following numbers in one's complement form. + 7 and – 7

   Ans: $(+7)_{10} = (0\,1\,1\,1)_2$, $(-7)_{10} = (1\,0\,0\,0)_2$
3. Convert $(247)_{10}$ into octal

   Ans: $(367)_8$
4. Convert $(736)_8$ into an equivalent binary number

   Ans: $(111\ 011\ 110)_2$
5. Convert $(3A.2F)_{16}$ to decimal

   Ans: $(58.1836)_{10}$
6. Convert $(0.\ 000\ 1111\ 0101\ 101)_2$ to Hexadecimal

   Ans: $(0.1EB4)_{16}$
7. Convert $(A72E)_{16}$ to octal

   Ans: $(123456)_8$

8. Subtract $(5C)_{16}$ from $(3F)_{16}$

   Ans: $(1D)_{16}$
9. Give an example of Alpha numeric code

   Ans: ASCII
10. Convert $(247.36)_8$ to equivalent hex number

    Ans: $(A7.78)_{16}$
11. Convert $(65535)_{10}$ to its binary and hexadecimal equivalents

    Ans: Binary → 1111 1111 1111 1111

    Hex → F F F F
12. Give any one use of ASCII code

    Ans: (i) To standardize computer hardware such as keyboards, printers and video displays.

    (ii) For sending digital data over telephone lines.
13. Write your full name in ASCII code. From table we can write

    Ans:
    $$\begin{array}{ccc} \underbrace{100\,1110}_{N} & \underbrace{100\,0001}_{A} & \underbrace{101\,0010}_{R} \\ \underbrace{100\,0001}_{A} & \underbrace{101\,1001}_{Y} & \underbrace{100\,0001}_{A} \\ \underbrace{100\,1110}_{N} & \underbrace{100\,0001}_{A} & \underbrace{100\,1110}_{N} \end{array}$$
14. Explain the conversion of decimal number into other number systems.

    Ans: This can be possible by sum of weights method or by repeated division by base b. Here in repeated division by base (example 2 in binary, 16 in Hex. etc.), the remainders are read from bottom to top.

    At the same time, the decimal fraction part of the decimal number is converted by using sum of weights method or by repeated multiplication by base b. Here the integers to the left of the radix point are read from top to bottom.
15. What is the difference between nibble and byte?

Ans: **Nibble** is a group of 4 binary bits. But **Byte** is a group of 8 bits.

16. Why Hexadecimal system is most popular?

Ans: It is the easiest way to convert large decimal numbers into binary.

17. Explain 'word' and 'word length'.

Ans: In a digital system, a group of bits processed at a time is called a **word**. But **word length** is a number of bits used to make a word.

18. What is the difference between **sign magnitude representation** and **1's complement representation**?

Ans: In a number the left most digit is the Most Significant Bit (MSB) and the right most digit is the Least Significant Bit (LSB). MSB is used to represent sign. 0 is used to represent +ve number and 1 is used to represent –ve number.

But in 1's complement form, +ve numbers are written as in the straight binary form. But the –ve numbers are written by subtracting equivalent positive number form $(2^n - 1)$ where n is the number of bits used.

19. Give 2's complement of 0

Ans: 0

20. How a –ve number can be converted into a +ve number?

Ans: By finding its 2's complement.

21. Explain '**end around carry**'.

Ans: It is the process of adding the carry bit to the LSB.

22. 8085 Microprocessor is a 8 bit microprocessor. Its wordlength is 8 bit. How large binary numbers are represented by its registers?

Ans: Using double precision (two storage locations or registers). Here registers are pairs of 8 bit each.

23. Explain 'sign-magnitude' of a representation?

Ans : Here an additional bit called the sign bit is placed in front of the number. If this bit is 0, the number is +ve. If it is a 1, the number is –ve.

24. In a code of base 7, the digits are 0, 1, 2, 3, 4, 5, 6. The number 468 is in decimal code. What is the equivalent in a code of base7? (CU Nov. 2016)

Ans :
$$\frac{468}{7} = 66.857 \longrightarrow 0.857\times 7 = 6$$
$$\frac{66}{7} = 9.429 \longrightarrow 0.429\times 7 = 3$$
$$\frac{9}{7} = 1.286 \longrightarrow 0.286\times 7 = 2$$
$$\frac{1}{7} = 0.143 \longrightarrow 0.143\times 7 = 1$$

$$(486)_{10} = \mathbf{(1236)_7}$$

25. Explain 1's complement method of binary subtraction with example. (CU Nov. 2016)

Example

Minuend	1111	
Subtrahend	–1011	
Here	1111	
	0100	(1s complement of 1011)
Adding	10011	
	$\rightarrow$ 1	
	0100	

$\therefore$ 1111 – 1011 $= (0100)_2 = (4)_{10}$

Another example

Minuend	110011	110011	
Subtrahend	–100101	+ 011010	(1s complement of 100101)
		1001101	
		$\rightarrow$ 1	
		1100	

$\therefore$ 110011 – 100101 $= (1110)_2 = (14)_{10}$

26. A computer is transmitting the following groups of bytes to an output device. Give the equivalent octal and hexadecimal listings 10110110, 01111011, 10010101, 00101110. (CU Nov. 2016)

Ans : The integer part of the binary number can be converted in to their octal equivalent by making groups of three bits starting form LSB and moving towards MSB and then replacing each group of three bits by its equivalent octal representation. (If the number of bits is less than three in the last group add zeros to the left side).

Binary equivalent octal

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

(i) $(10110110)_2 = 010 \quad 110 \quad 110$

$= 2 \quad 6 \quad 6$

$= (266)_8$

(ii) $(01111011)_2 = (173)_8$

(iii) $(10010101)_2 = (225)_8$

(iv) $(00101110)_2 = (056)_8$

The integer part of a binary number can be converted in to its hexadecimal equivalent by making group of four bits starting from LSB and moving forwards MSB and then replacing each group of four bits by its equivalent hexadecimal representation. (If the number of bits is less than four in the last group add zeros to the left side).

(i) $(10110110)_2 = 1011 \quad 0110$

$= (B\ 6)_{16}$

(ii) $(01111011)_2 = (7\ B)_{16}$

(iii) $(10010101)_2 = (95)_{16}$

(iv) $(00101110)_2 = (2\ E)_{16}$

Assignments

1. Convert $[256.72]_8$ to binary.
2. Convert $[11101.101101]_2$ to octal.
3. Convert $[111011011]_2$ to hexadecimal
4. Convert $[11011]_2$ to decimal

   Ans: 27
5. Convert $[0.110]_2$ to decimal

   Ans: 0.75
6. Convert $[101101.101]_2$ to decimal

   Ans: 45.625
7. Convert $[F8\ E6.39]_H$ to decimal
8. Convert $[62359]_{10}$ to Hexadecimal.
9. What is 2's complement representation?
10. What is the base or radix of Binary number system?

    Ans: 2
11. What is the base of decimal system?

    Ans: 10
12. What is the radix of Hexadecimal system

    Ans: 16

13. What is a bit?

    Ans: Binary digit. (0 or 1)

14. Find the equivalent binary number for $(48)_{10}$.

15. Convert $(1101)_2$ to decimal?

    Ans: $(13)_{10}$.

16. Find a 1 byte representation of – 9 using the 2's complement method.

17. Convert $(7502)_8$ to decimal.

    Ans: $(3906)_{10}$

18. Convert $(FCB)_{16}$ to binary.

    Ans: 1111 1100 1011

19. Convert 0111 1011 1101 to Hexadecimal

    Ans: $(7BD)_{16}$

20. Convert $(AB\ 60)_{16}$ to decimal

21. Convert $(256)_{10}$ to hexadecimal

22. What is EBCDIC code?

23. What is ASCII code?

24. Convert $(101\ AH)_{16}$ to decimal.

25. What is under flow?